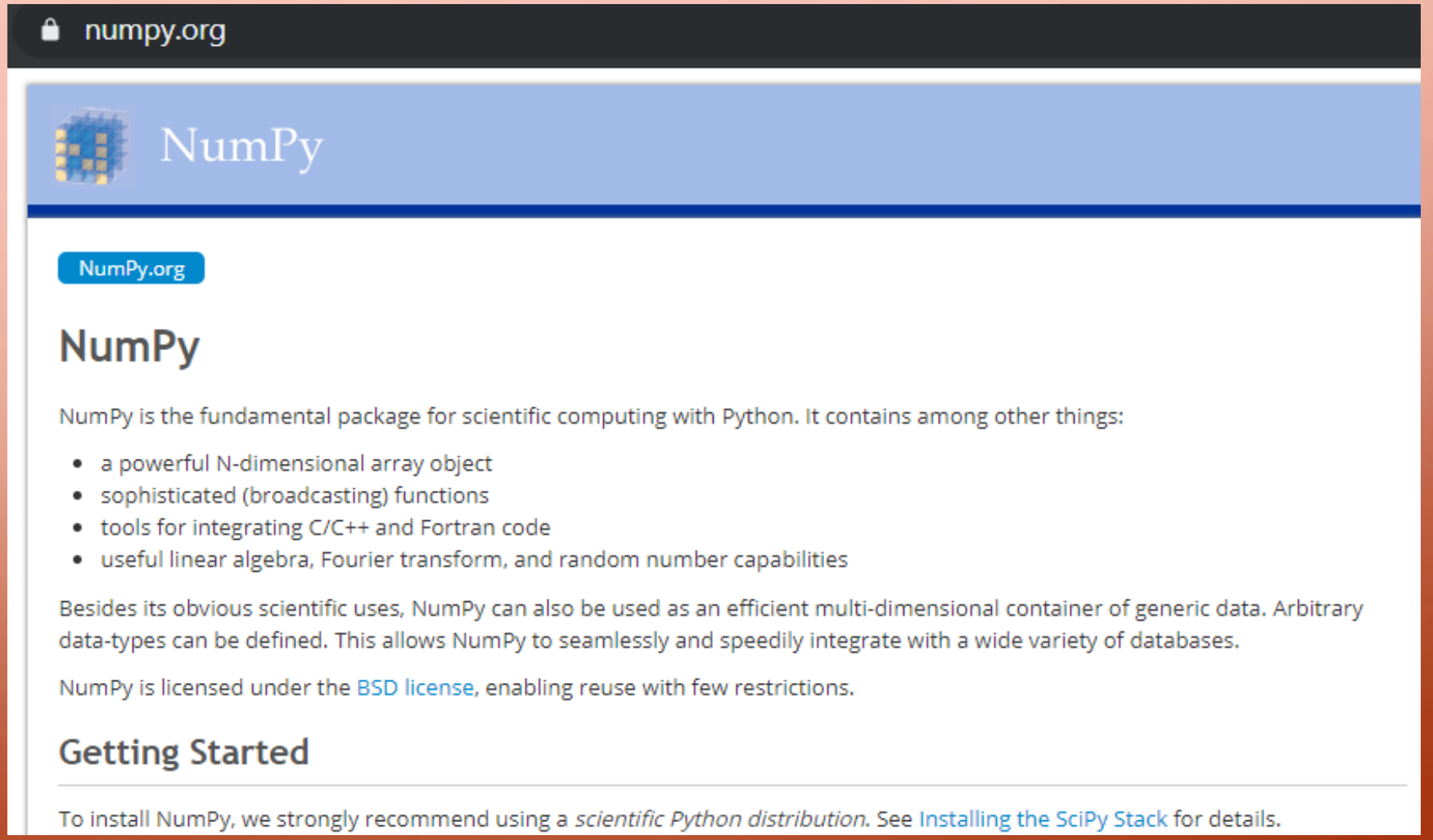
PROGRAMANDO CON LIBRERÍAS DE PYTHON

NUMPY


numpy.org
NumPy
NumPy.org
NumPy
NumPy is the fundamental package for scientific computing with Python. It contains among other things:
a powerful N-dimensional array object
sophisticated (broadcasting) functions
tools for integrating C/C++ and Fortran code
useful linear algebra, Fourier transform, and random number capabilities
Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.
NumPy is licensed under the BSD license, enabling reuse with few restrictions.
Getting Started
To install NumPy, we strongly recommend using a scientific Python distribution. See Installing the SciPy Stack for details.

MATPLOTLIB

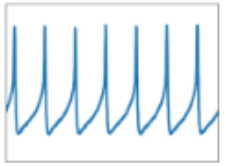
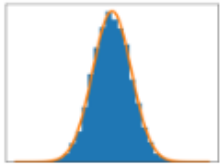
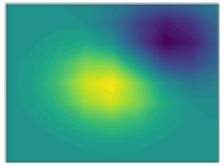
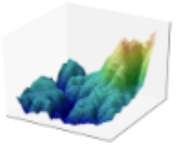
matplotlib.org

matplotlib

Version 3.1.1

home | examples | tutorials | API | contents »

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.

Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

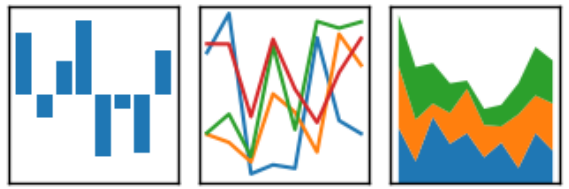
For simple plotting the `pyplot` module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

PANDAS

pandas.pydata.org

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



home // about // get pandas // documentation // community // talks // donate

Python Data Analysis Library

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

pandas is a NumFOCUS sponsored project. This will help ensure the success of development of *pandas* as a world-class open-source project, and makes it possible to donate to the project.

A Fiscally Sponsored Project of

NUMFOCUS

OPEN CODE = BETTER SCIENCE

VERSIONS

Release
0.25.1 - August 2019
download // docs // pdf

Development
0.26.0 - September 2019
github // docs

Previous Releases
0.25.0 - download // docs // pdf
0.24.2 - download // docs // pdf
0.24.1 - download // docs // pdf
0.24.0 - download // docs // pdf
0.23.4 - download // docs // pdf

NUMPY

NUMPY

- Crear un array de 15 elementos

```
import numpy as np

a = np.arange(15)

print(a)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10
11 12 13 14]
```

NUMPY

- Crear un array de 15 elementos
- Convertir en matriz de dos dimensiones (3x5)

```
a = np.arange(15)

a = a.reshape(3,5)

print(a)
```

```
[[ 0 1 2 3 4]

[ 5 6 7 8 9]

[10 11 12 13 14]]
```

NUMPY

- Acceder a datos específicos de una matriz

```
[[ 0 1 2 3 4]
[ 5 6 7 8 9]
[10 11 12 13 14]]
```

```
print(a[2,3])
13
print(a[2,1:6])
[11 12 13 14]
```

NUMPY

- Array con tipos de datos entero o flotante

```
b = np.array([2,3,4])
b.dtype
dtype('int64’)
b =
np.array([2.5,4.6,8.7])
b.dtype
dtype('float64')
```

NUMPY

- Array de unos o ceros

```
c = np.zeros((3,4))
print(c)
[[0. 0. 0. 0.]
[0. 0. 0. 0.]
[0. 0. 0. 0.]]
d = np.ones((4,5))
print(d)
[[1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
[1. 1. 1. 1. 1.]]
```

NUMPY

- Suma de dos matrices

- $a = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}, b = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$

```
a = np.array([[1,1],[1,2]])

b = np.array([[2,0],[0,1]])
```

?

NUMPY

- Suma de dos matrices
- $a = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}, b = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$

```
print(a+b)
```

```
[[3 1]
[1 3]]
```

NUMPY

- Multiplicación de dos matrices
- $a = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}, b = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$

```
print(a*b)
```

RESULTADO ?

NUMPY

- Multiplicación de dos matrices
- $a = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}, b = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$

```
print(a*b)
```

```
[[2 0]
[0 2]]
```

NUMPY

- Producto escalar de dos matrices
- $a = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}, b = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$

```
print(a @ b)
```

```
[[2 1]
[2 2]]
```

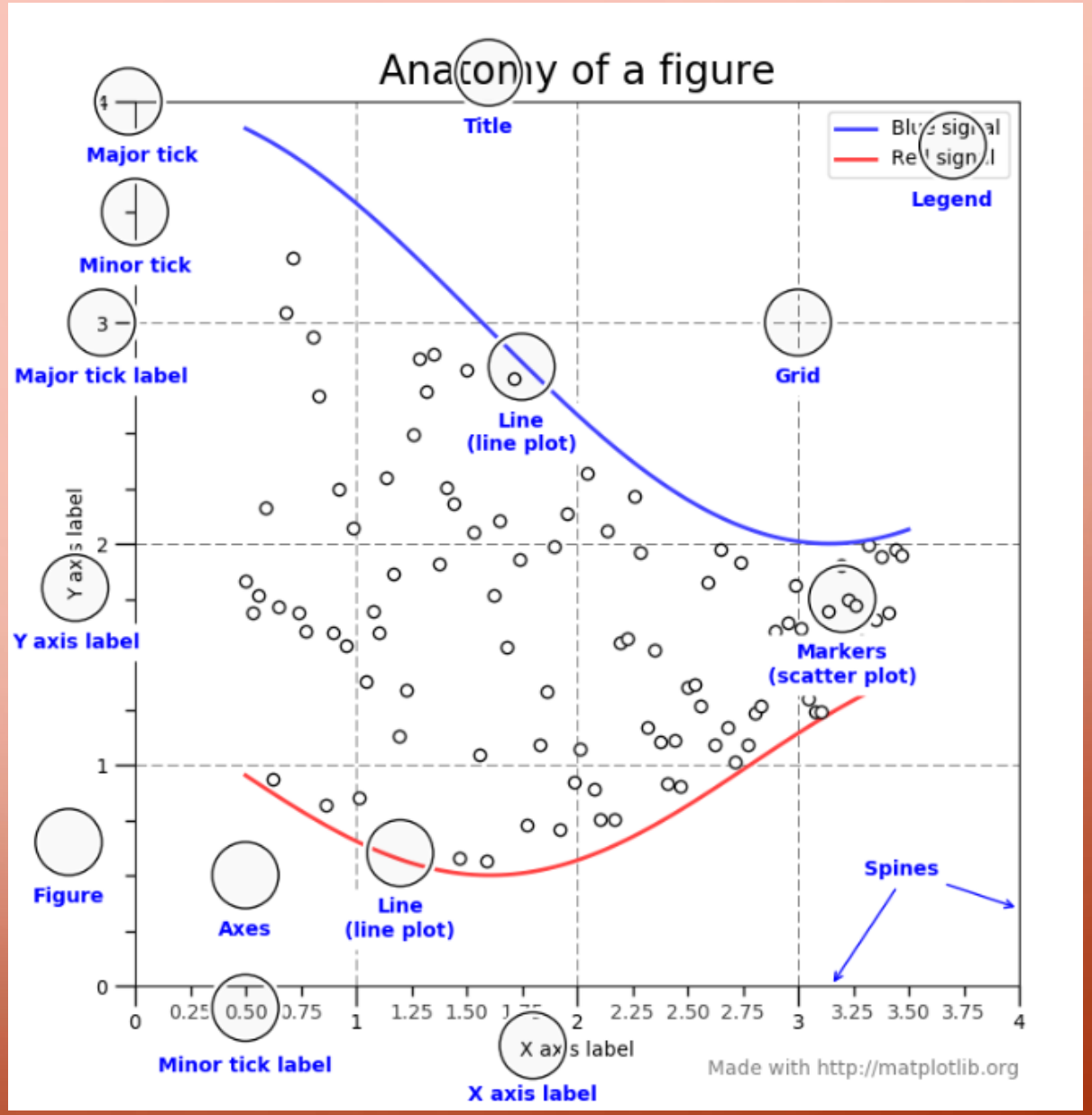
NUMPY

- $a = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}, b = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$
- Sumar, mínimo, máximo, ordenar …

```
a.sum()
5
a.min()
1
a.max()
2
b.sort()
print(b)
[[0 2]
 [0 1]]
```

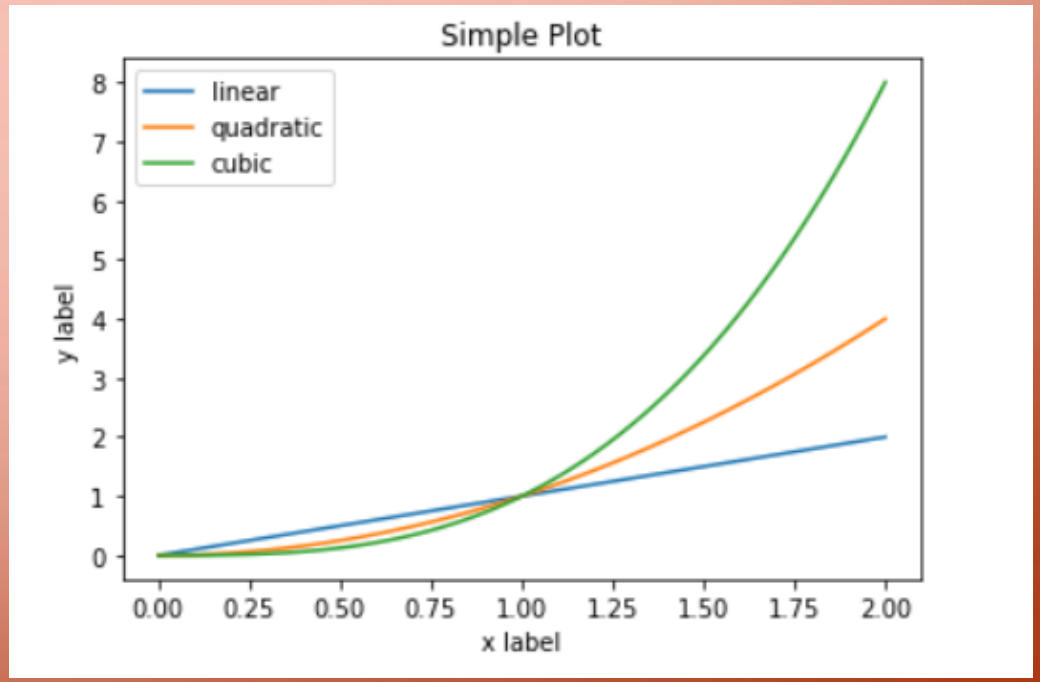
MATPLOTLIB

MATPLOTLIB


Anatomy of a figure
Title
Major tick
Minor tick
Major tick label
Legend
Grid
Line
(line plot)
Y axis label
Markers
(scatter plot)
Figure
Axes
Line
(line plot)
Spines
Minor tick label
X axis label
Made with http://matplotlib.org

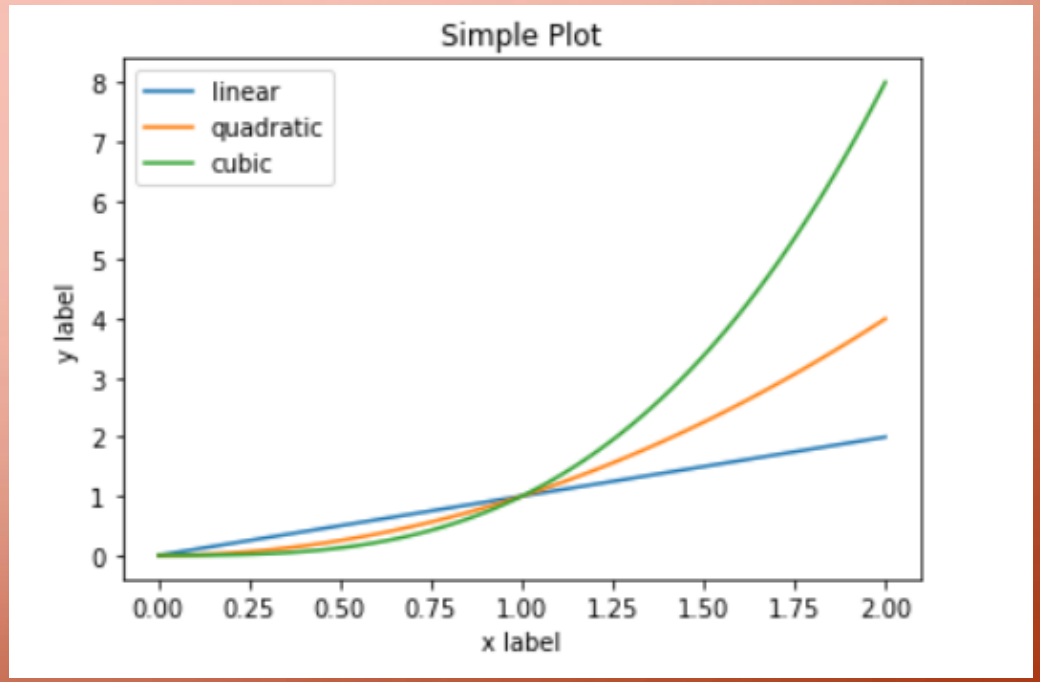
MATPLOTLIB

```
x = np.linspace(0, 2, 100)


plt.plot(x, x, label='linear')

plt.plot(x, x**2, label='quadratic’)

## FUNCION CUBICA


plt.xlabel('x label')

plt.ylabel('y label')


plt.title("Simple Plot")

plt.legend() # muestra los label

plt.show()
```



MATPLOTLIB

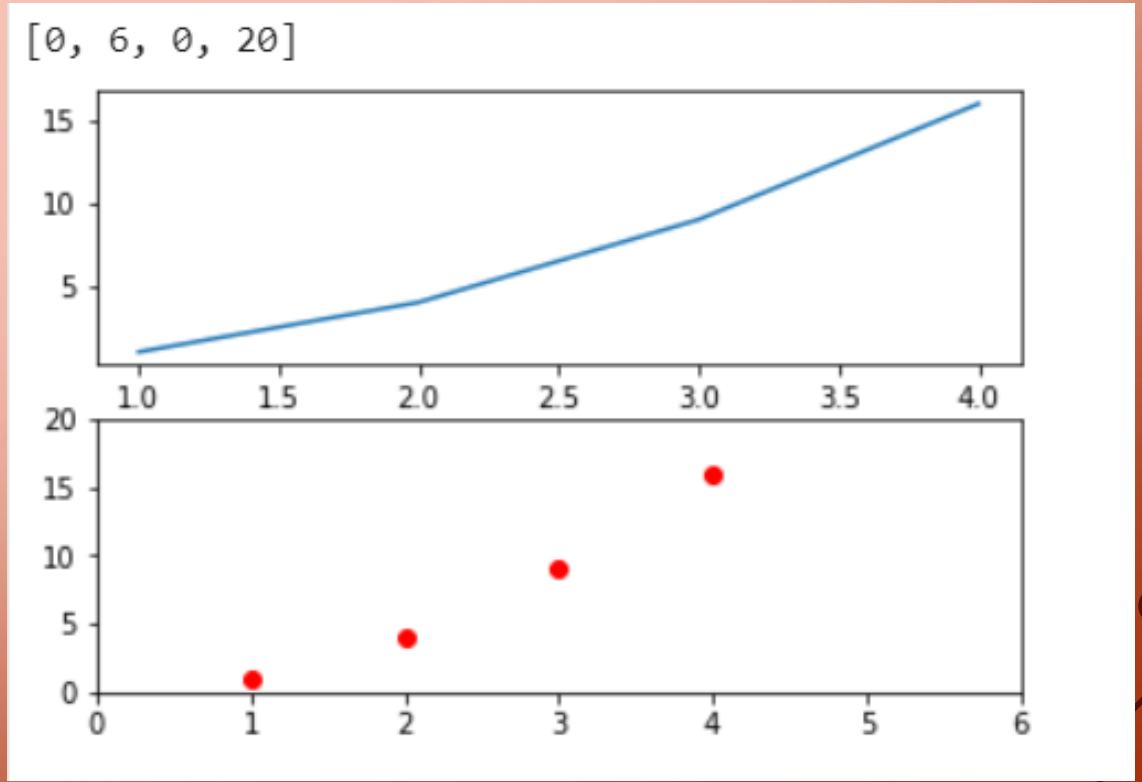
```
x = np.linspace(0, 2, 100)


plt.plot(x, x, label='linear')

plt.plot(x, x**2, label='quadratic')

plt.plot(x, x**3, label='cubic')


plt.xlabel('x label')

plt.ylabel('y label')


plt.title("Simple Plot")

plt.legend() # muestra los label

plt.show()
```



MATPLOTLIB

MÚLTIPLES GRÁFICOS

```
plt.subplot(2,1,1)

plt.plot([1, 2, 3, 4], [1, 4, 9, 16])


plt.subplot(2,1,2)

plt.plot([1, 2, 3, 4], [1, 4, 9, 16],
'ro')

# <color><forma -- s ^>

plt.axis([0, 6, 0, 20])
```

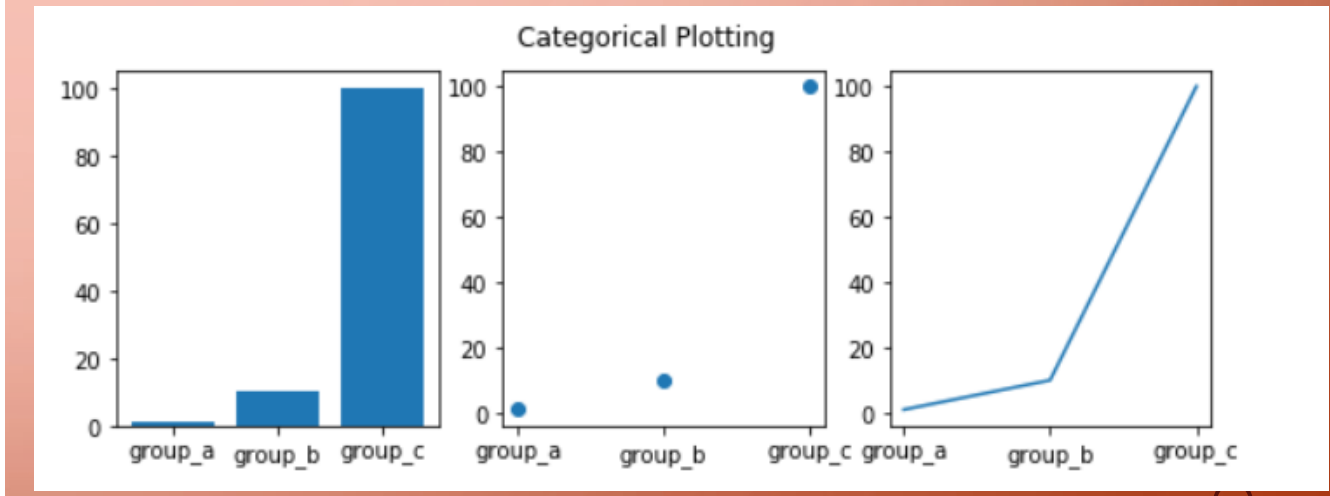


MATPLOTLIB

- Como mostrar 4 gráficos?

MATPLOTLIB

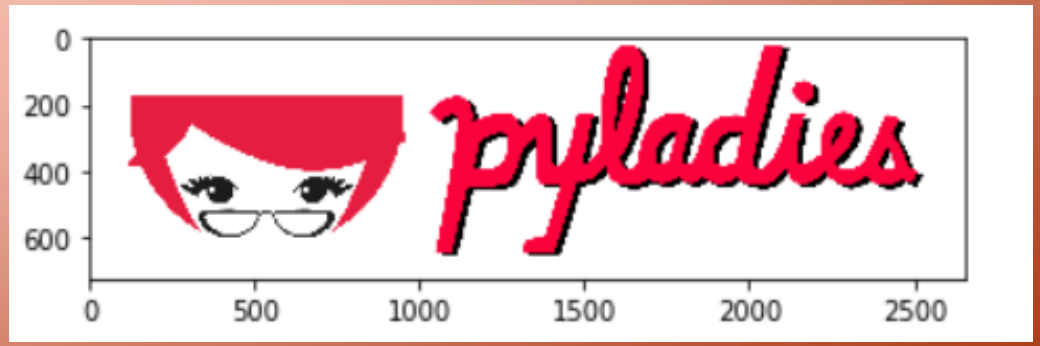
```
names = ['group_a', 'group_b', 'group_c']
values = [1, 10, 100]

plt.figure(figsize=(9, 3))

plt.subplot(131)
plt.bar(names, values)
plt.subplot(132)
plt.scatter(names, values)
plt.subplot(133)
plt.plot(names, values)
plt.suptitle('Categorical Plotting')
plt.show()
```



MATPLOTLIB

SE PUEDE MOSTRAR IMÁGENES

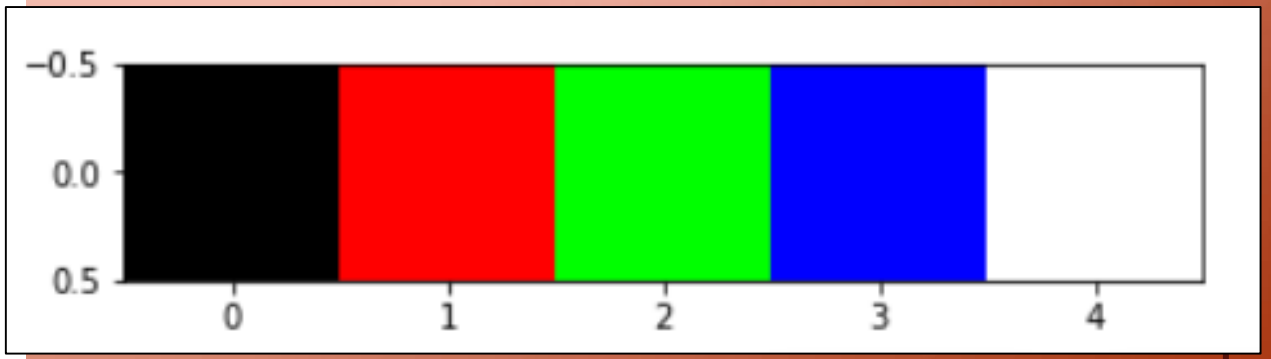
```
a =
plt.imread("https://raw.github
usercontent.com/pyladies-
bcn/pyladies_latex_template/ma
ster/pyladies.png")

plt.imshow(a)

plt.show()
```



MATPLOTLIB

SE PUEDE MOSTRAR IMÁGENES

```
image = np.array(

[[[0,0,0],

[255,0,0],

[0,255,0],

[0,0,255],

[255,255,255]]]

)
plt.imshow(image)

plt.show()
```



PANDAS

PANDAS

- Manipular datos CSV

	A	B	C	D	E	F	G	H	I	J
1	age	Workclass	fnlwgt	education	education_nu	marital_statu	occupation	relationship	race	sex
2	47	'Private'	51835	'Prof-school'	15	'Married-civ-s	'Prof-specialty	'Wife'	'White'	'Female'
3	50	'Federal-gov'	251585	'Bachelors'	13	'Divorced'	'Exec-manage	'Not-in-family	'White'	'Male'
4	47	'Self-emp-inc'	109832	'HS-grad'	9	'Divorced'	'Exec-manage	'Not-in-family	'White'	'Male'
5	43	'Private'	237993	'Some-college	10	'Married-civ-s	'Tech-support	'Husband'	'White'	'Male'
6	46	'Private'	216666	'5th-6th'	3	'Married-civ-s	'Machine-op-i	'Husband'	'White'	'Male'
7	35	'Private'	56352	'Assoc-voc'	11	'Married-civ-s	'Other-service	'Husband'	'White'	'Male'
8	41	'Private'	147372	'HS-grad'	9	'Married-civ-s	'Adm-clerical'	'Husband'	'White'	'Male'
9	30	'Private'	188146	'HS-grad'	9	'Married-civ-s	'Machine-op-i	'Husband'	'White'	'Male'
10	30	'Private'	59496	'Bachelors'	13	'Married-civ-s	'Sales'	'Husband'	'White'	'Male'
11	32	'<undefined>'	293936	'7th-8th'	4	'Married-spou	'<undefined>'	'Not-in-family	'White'	'Male'
12	48	'Private'	149640	'HS-grad'	9	'Married-civ-s	'Transport-mo	'Husband'	'White'	'Male'
13	42	'Private'	116632	'Doctorate'	16	'Married-civ-s	'Prof-specialty	'Husband'	'White'	'Male'
14	29	'Private'	105598	'Some-college	10	'Divorced'	'Tech-support	'Not-in-family	'White'	'Male'
15	36	'Private'	155537	'HS-grad'	9	'Married-civ-s	'Craft-repair'	'Husband'	'White'	'Male'

PANDAS

```
import pandas as pd

datos = pd.read_csv('smallSet.csv')

print(datos[1:10])
```

```
   age        Workclass  fnlwgt  ... hours_per_week   native_country   salary
1   50    'Federal-gov'  251585  ...             55  'United-States'    '>50K'
2   47  'Self-emp-inc'  109832  ...             60  'United-States'  '<=50K'
3   43        'Private'  237993  ...             40  'United-States'    '>50K'
4   46        'Private'  216666  ...             40         'Mexico'  '<=50K'
5   35        'Private'   56352  ...             40    'Puerto-Rico'  '<=50K'
6   41        'Private'  147372  ...             48  'United-States'  '<=50K'
7   30        'Private'  188146  ...             40  'United-States'  '<=50K'
8   30        'Private'   59496  ...             40  'United-States'  '<=50K'
9   32   '<undefined>'  293936  ...             40   '<undefined>'  '<=50K'

[9 rows x 15 columns]
```

PANDAS

- Extraer una columna de datos

```
[ ] print(datos['native_country'])
```

```
0          'Honduras'
1     'United-States'
2     'United-States'
3     'United-States'
4            'Mexico'
5       'Puerto-Rico'
6     'United-States'
7     'United-States'
8     'United-States'
9       '<undefined>'
10    'United-States'
11    'United-States'
12    'United-States'
13    'United-States'
14    'United-States'
15    'United-States'
16    'United-States'
17    'United-States'
18    'United-States'
19    'United-States'
20    'United-States'
21    'United-States'
22    'United-States'
23           'Mexico'
24    'United-States'
```

PANDAS

- Mostrar 5 primeros registros

```
[ ] datos.head()
```

	age	Workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	c
0	47	'Private'	51835	'Prof-school'	15	'Married-civ-spouse'	'Prof-specialty'	'Wife'	'White'	'Female'	
1	50	'Federal-gov'	251585	'Bachelors'	13	'Divorced'	'Exec-managerial'	'Not-in-family'	'White'	'Male'	
2	47	'Self-emp-inc'	109832	'HS-grad'	9	'Divorced'	'Exec-managerial'	'Not-in-family'	'White'	'Male'	
3	43	'Private'	237993	'Some-college'	10	'Married-civ-spouse'	'Tech-support'	'Husband'	'White'	'Male'	
4	46	'Private'	216666	'5th-6th'	3	'Married-civ-spouse'	'Machine-op-inspct'	'Husband'	'White'	'Male'	

PANDAS

- Mostrar registros aleatorios

```
[ ]  datos.sample(2)
```

	age	Workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex
21	23	'Private'	211678	'Some-college'	10	'Never-married'	'Machine-op-inspct'	'Not-in-family'	'White'	'Male'
40	28	'Private'	212563	'Some-college'	10	'Divorced'	'Machine-op-inspct'	'Unmarried'	'Black'	'Female'

PANDAS

- Obtener datos filtrados por un valor de una columna

```
[ ]  datos[datos.sex == "'Female'"]
```

	age	Workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	c
0	47	'Private'	51835	'Prof-school'	15	'Married-civ-spouse'	'Prof-specialty'	'Wife'	'White'	'Female'	0	
14	28	'Private'	183175	'Some-college'	10	'Divorced'	'Adm-clerical'	'Not-in-family'	'White'	'Female'	0	
15	53	'Private'	169846	'HS-grad'	9	'Married-civ-spouse'	'Adm-clerical'	'Wife'	'White'	'Female'	0	
19	31	'Private'	309974	'Bachelors'	13	'Separated'	'Sales'	'Own-child'	'Black'	'Female'	0	
26	18	'Private'	309634	'11th'	7	'Never-married'	'Other-service'	'Own-child'	'White'	'Female'	0	
30	46	'Private'	51618	'HS-grad'	9	'Married-civ-spouse'	'Other-service'	'Wife'	'White'	'Female'	0	
32	44	'Private'	343591	'HS-grad'	9	'Divorced'	'Craft-repair'	'Not-in-family'	'White'	'Female'	14344	

PANDAS

- Análisis de datos numéricos en columnas

```
datos.age.max()

79

datos.age.mean()

39.208333333333336
```

PANDAS

ANÁLISIS DE DATOS NUMÉRICOS EN COLUMNAS

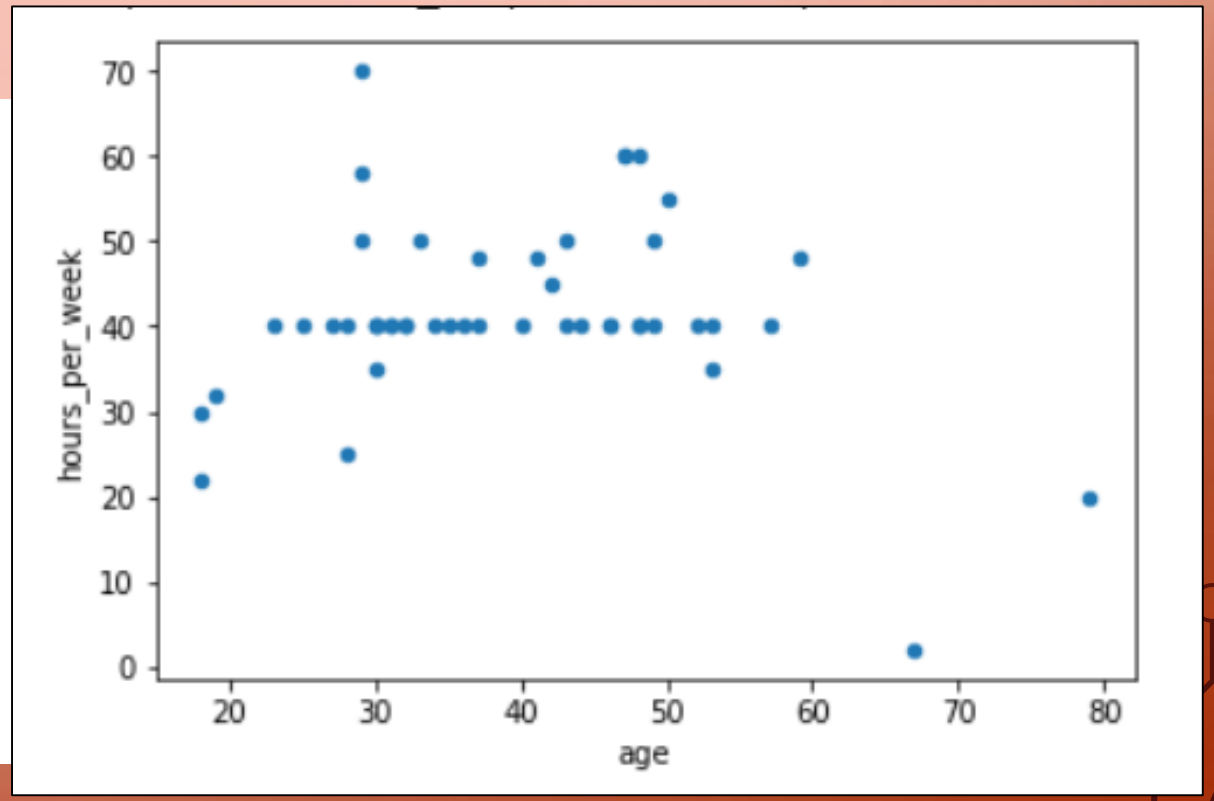
```
desc = pd.DataFrame(datos.age)

print(desc.describe())
```

```
                age
count     48.000000
mean      39.208333
std       12.709451
min       18.000000
25%       30.000000
50%       37.000000
75%       48.000000
max       79.000000
```

PANDAS

GRÁFICOS ENTRE COLUMNAS

```
df = pd.DataFrame(datos,
columns=["age","hours_per_week"]
)

df.plot.scatter(x='age',
y='hours_per_week')
```



REGRESIÓN LINEAL USANDO LAS LIBRERÍAS

- Se procederá a realizar una regresión lineal entre dos columnas de datos

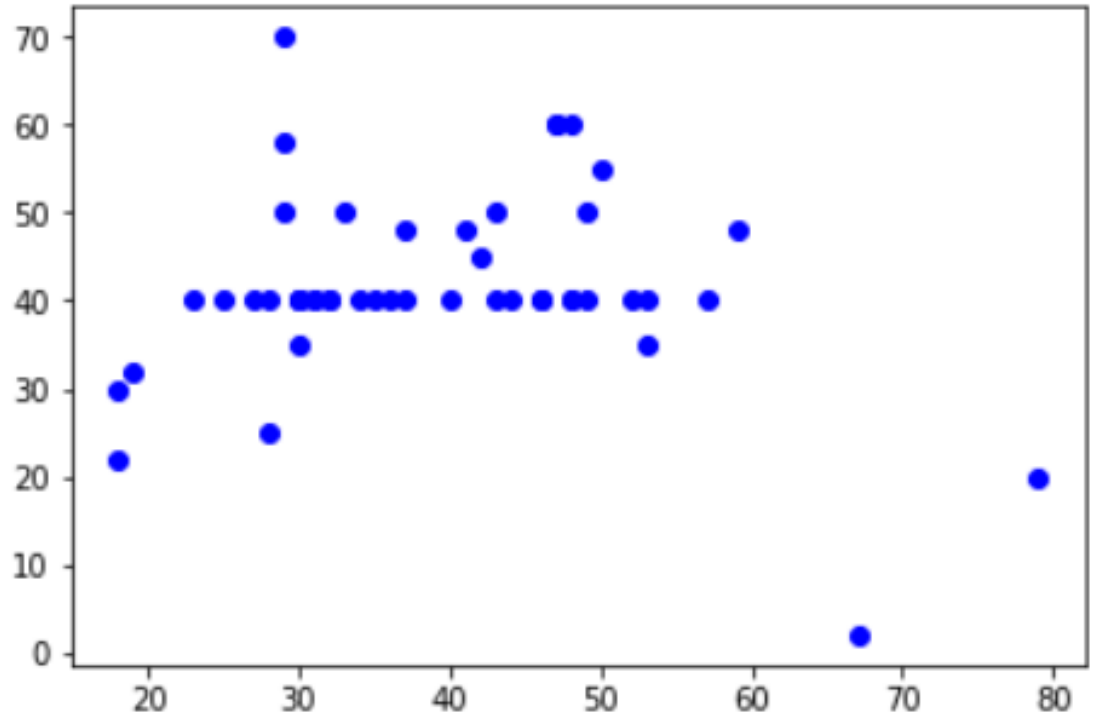
- `x_data = datos["age"]`
- `y_data = datos["hours_per_week"]`

REGRESIÓN LINEAL USANDO LAS LIBRERÍAS

- Visualización de los datos

```
plt.scatter(x_data, y_data, color='blue')
```

```
<matplotlib.collections.PathCollection at 0x7f459ea5dba8>
```



REGRESIÓN LINEAL USANDO LAS LIBRERÍAS

- y = wx + b

```
from sklearn.linear_model import LinearRegression

x_data = np.array(x_data)
x_data_array = x_data.reshape(-1,1)

reg_lineal = LinearRegression() # creamos una instancia de LinearRegression

# instruimos a la regresión lineal que aprenda de los datos (x,y)
reg_lineal.fit(x_data_array, y_data)

# vemos los parámetros que ha estimado la regresión lineal
print('w = ' + str(reg_lineal.coef_) + ', b = ' + str(reg_lineal.intercept_))

>>> w = [-0.08577654], b = 44.88398845263054
```

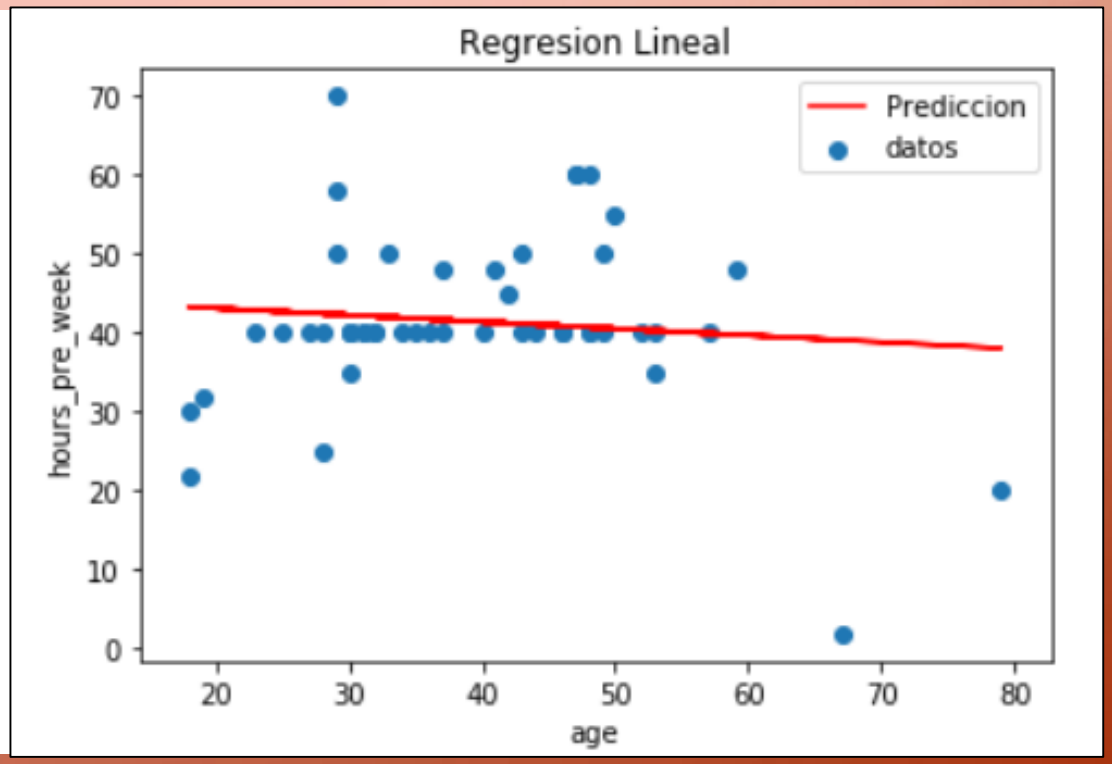
REGRESIÓN LINEAL USANDO LAS LIBRERÍAS

```
y_pred = reg_lineal.predict(x_data_array)
print(y_pred)
```

```
[40.85249114 40.59516152 40.85249114 41.19559729 40.93826768 41.8818096
 41.36715037 42.31069229 42.31069229 42.13913922 40.7667146  41.28137383
 42.39646883 41.79603306 42.48224537 40.3378319  40.68093806 42.73957499
 43.25423422 42.22491575 42.39646883 42.91112806 38.1076419  42.56802191
 41.45292691 39.13696036 43.34001076 42.22491575 43.34001076 40.42360844
 40.93826768 39.82317267 41.10982075 40.3378319  40.68093806 42.05336268
 42.31069229 41.19559729 39.99472575 41.71025652 42.48224537 42.31069229
 41.96758614 42.39646883 40.7667146  41.71025652 40.7667146  42.13913922]
```

REGRESIÓN LINEAL USANDO LAS LIBRERÍAS

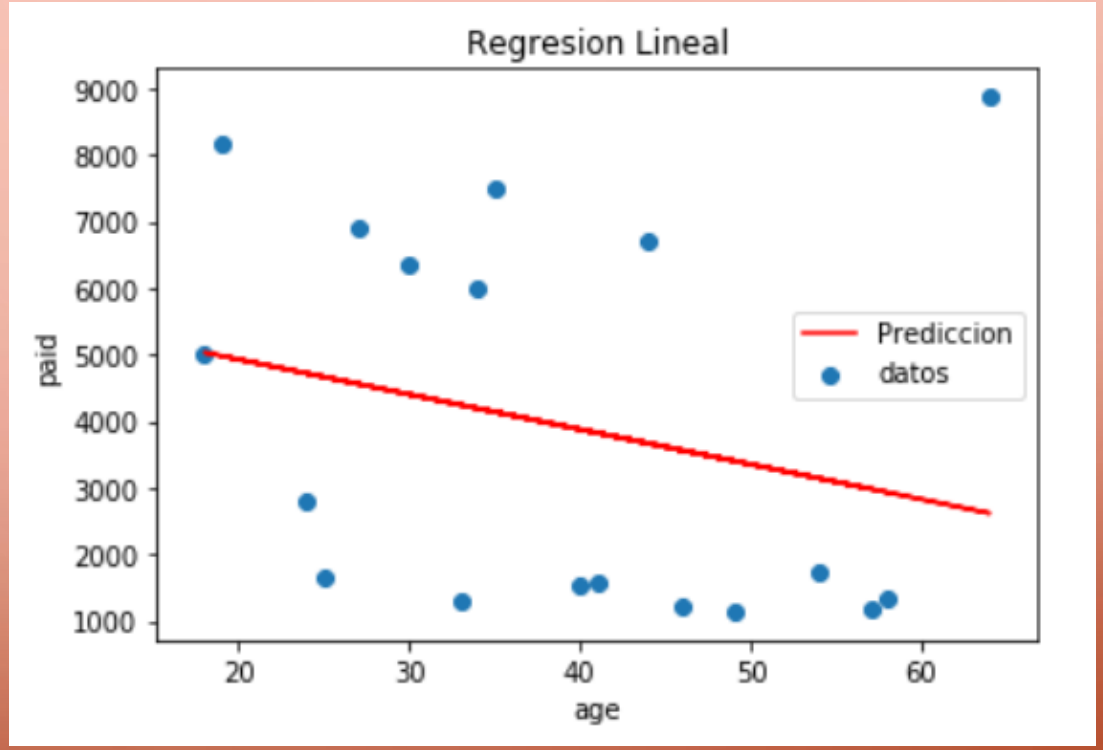
```
plt.plot(x_data, y_pred, color='red' , label='Prediccion')

plt.scatter(x_data,y_data, label='datos')


plt.title('Regresion Lineal')

plt.xlabel('age')

plt.ylabel('hours_pre_week')

plt.legend()

plt.show()
```



Gracias

TAREA

- Mostrar la relación entre pago y edad en mujeres menores de 65


Regresion Lineal
paid
age
Prediccion
datos
9000
8000
7000
6000
5000
4000
3000
2000
1000
20
30
40
50
60